Roadrunner Technical Seminar Series

Overview of Modeling, Performance, and Results

March 19th 2008

Kevin J. Barker, Kei Davis, Adolfy Hoisie, Michael Lang, Scott Pakin, Jose Sancho-Pitarch

Presented by: Darren J. Kerbyson

Performance and Architecture Laboratory (PAL)

http://www.c3.lanl.gov/pal

Computer, Computational & Statistical Sciences Division

LA-UR 08-2037

PAL Performance and Architecture Lab

- **Novel techniques developed by PAL at Los Alamos**
  - Methods are quasi-analytical
  - Models encapsulate performance of entire apps on full systems
- **The workload considered is diverse (ASC, SC, DARPA, NSF)**
- **Analyze existing systems (or near-market systems)**
  - Models validated on most large systems in the last decade
- **Examine possible future systems**
  - Design space exploration
- **Recent work includes:**
  - Roadrunner (>1Pf peak, Opteron + Cell-eDP @ Los Alamos)
  - IBM PERCS (DARPA HPCS, NSF track-1 @ NCSA ~2010)
  - Comparison of Red Storm, ASC Purple, BlueGene/L (SC’06)
  - Application modeling (ASC, DARPA, Office of Science)
- **Models are our tools for performance analysis.**
- **Models are predictive, and highly accurate**

PAL’s performance analysis of Roadrunner

Aug ’05: “Analysis of a two-level heterogeneous processing system”, (UCAS-2, Austin, TX March ’06)

Sept ’06: PAL RR report #1: Voltaire Switch Cabling Performance Issues

Oct ’06: PAL RR report #2: Application Specific Optimization of Infiniband Networks

Jan ’07: PAL RR report #3: Performance Acceptance Testing of Roadrunner Phase 1 (Single CU testing)

July ’07: PAL RR report #4: Early Performance Testing of the eDP version of the Cell-BE

Sep ’07: PAL RR report #5: A note on Application Performance of the eDP version of the Cell

Oct ’07: Presented performance analysis at RR assessments

- **On-going:**
  - benchmarking and modeling of actual system, Cell-Messaging Layer, JumboMem …

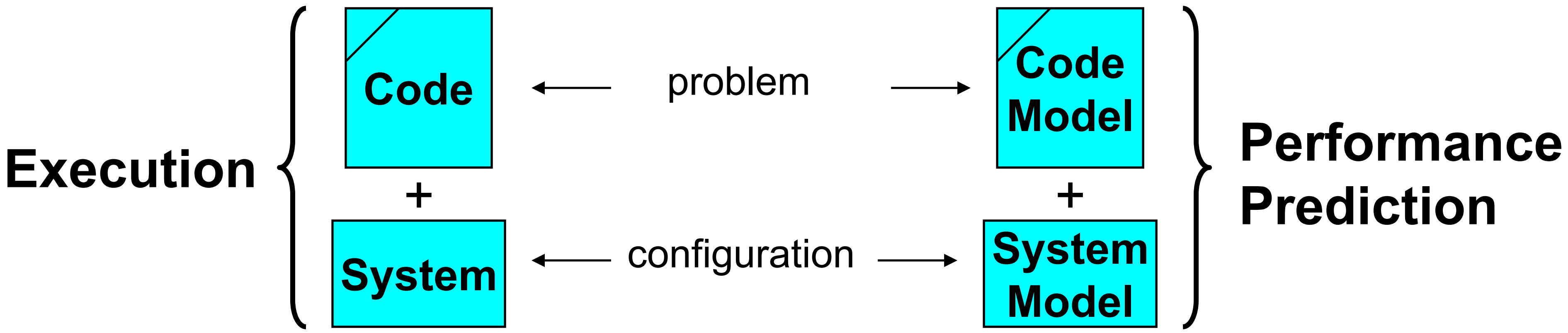
Talk Outline

- **Performance modeling methodology**
- **Architecture and performance parameters review**
- **Application performance**
    - VPIC
    - SPaSM
    - Sweep3D
    - Milagro
- **Performance prediction at scale**
- **Comparisons with other systems**

- **Note: Most of this analysis was undertaken in Aug/Sept ’07**
    - Many of the codes have progressed
    - System performance characteristics firming up
    - No measurement on actual hardware yet (imminent)

Question: How do we analyze the performance of a non-existent Machine?

- **Answer: Need a model.**
- **A model should encapsulate the understanding of:**
  - What resources an application uses during execution
  - How often it does it
  - How its usage changes when scaling
  - How long the system takes in order to satisfy the resource requirements



Execution

Code ← problem → Code Model

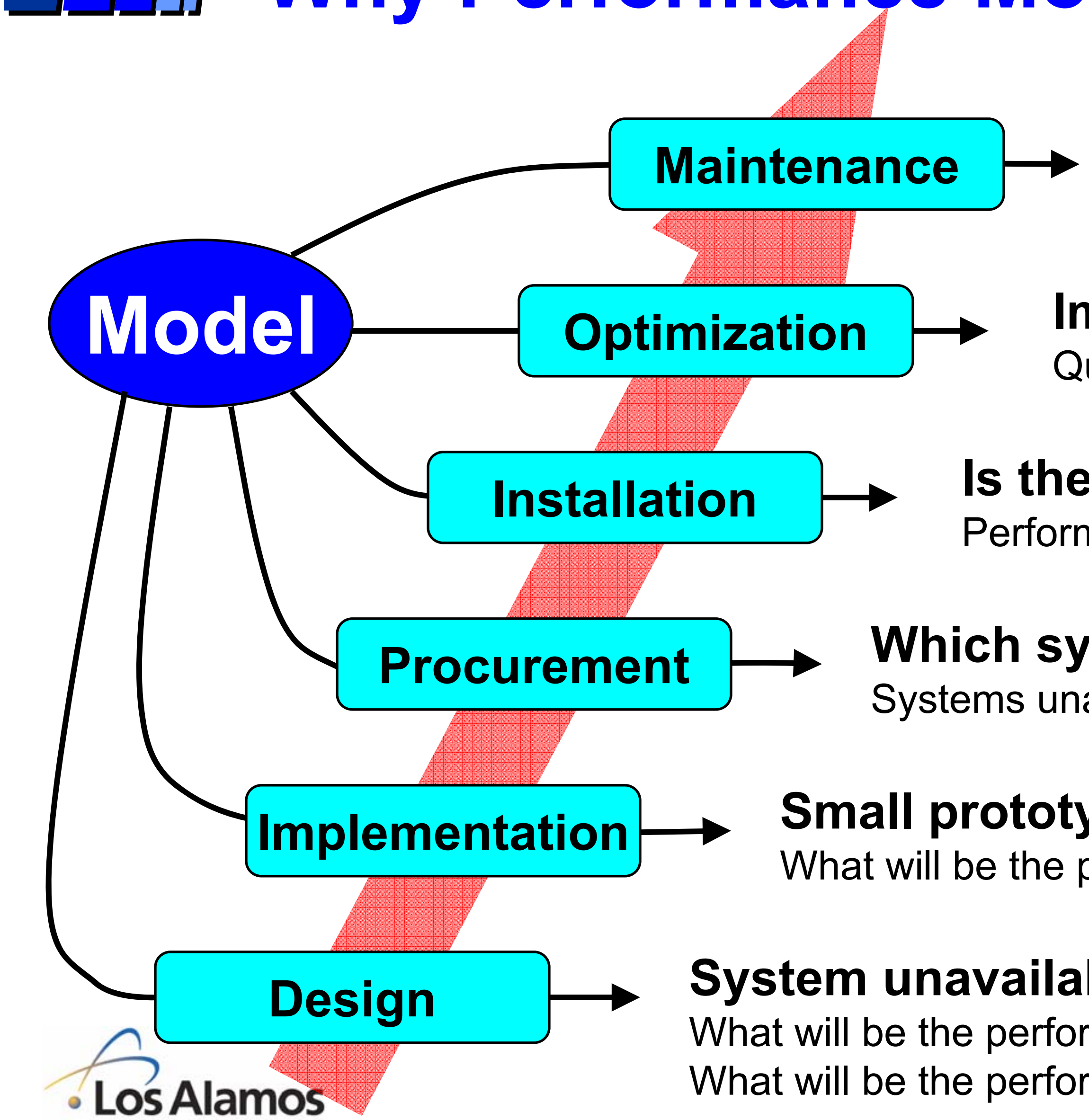
+ \+

System ← configuration → System Model

Performance Prediction

- **Application centric view – what the application doing**

Why Performance Modeling?



System updates
Quantify impact on performance

Improvements
Quantify impacts prior to implementation

Is the machine working?
Performance should be expected (Q, purple)

Which system should I buy?
Systems unavailable to measure (e.g. Roadrunner)

Small prototype available
What will be the performance of a 1.4PFlop system?

System unavailable for measurement
What will be the performance of IBM PERCS be in 2010?
What will be the performance of BG/P?

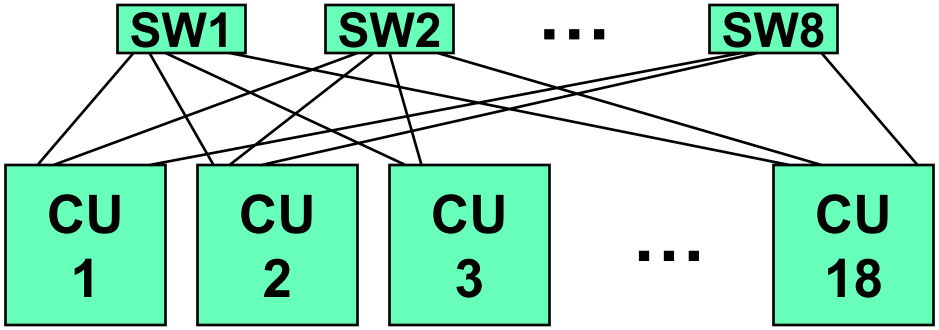
Diversity of Applications

- **1) VPIC**
  - Cell-centric, Opterons used only for Message relay
- **2) SPaSM**
  - Hybrid, Both Cell and Opterons do useful work
- **3) Sweep3D**
  - Cell-centric, Opterons used only for Message relay
- **4) Milagro**
  - 2 versions

- **For each:**
  - Examine computation, communication, and possible overlap
  - Use input-decks of interest
  - Develop performance model using existing systems
  - Validate model on existing systems
  - Use models to predict for RR

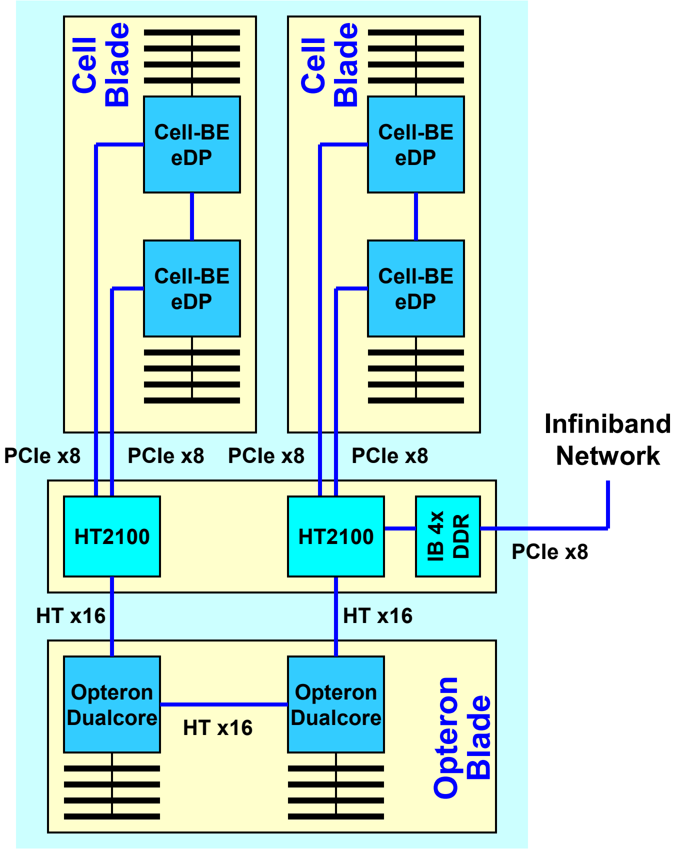
Essential System Peak Performance Parameters

- System = 18 CU = 3240 triblades
  = 12960 (AMD cores + cell eDP)
- Interconnected using Infiniband 4x DDR
  - Full fat-tree within a CU
  - 2:1 (reduced) fat-tree between CUs



- Peak DP flops = 1.4Pf/s
- Each CU contains 180 compute-nodes, 12 I/O-nodes

Essential Node (Triblade) Peak Performance Parameters



- **4 Cell eDP = 4x (PPU + 8 SPUs)**
  - Cell eDP = 104 Gflop/s (DP)
= 208 Gflop/s (SP)
- **4 AMD cores**
  - AMD = 3.6 Gflop/s (DP) / core
- **Cell <-> AMD**
  - Bandwidth = 2.0GB/s + 2.0GB/s
  - Latency ~1.5µs
- **AMD <-> AMD (inter-node)**
  - Bandwidth = 2.0GB/s + 2.0GB/s
  - Latency ~ 1.5µs

Data Movement Performance Characteristics of RR: Input to Models

		Worst	Probable	Best
Single Cell -> Opteron (uni)	Latency	4.5us	3us	1.5us
	Bandwidth	1.2GB/s	1.4GB/s	1.6GB/s
All cells -> Opteron (uni)	Latency	5.5us	4us	2.5us
	Bandwidth	1.1GB/s	1.3GB/s	1.5GB/s
Single Cell -> Opteron (Bi)	Latency	5.5us	4us	3.5us
	Bandwidth	1GB/s	1.2GB/s	1.4GB/s
All cells -> Opteron (Bi)	Latency	6.5us	5us	3.5us
	Bandwidth	0.9GB/s	1.1GB/s	1.3GB/s
Infiniband (Uni)	Latency	2.2us	2.0us	1.8us
	Bandwidth	1.3GB/s	1.5GB/s	1.7GB/s
Infiniband (Bi)	Latency	2.7us	2.5us	2.3us
	Bandwidth	1.2GB/s	1.4GB/s	1.6GB/s

NB. Measurement on actual RR Triblades is imminent

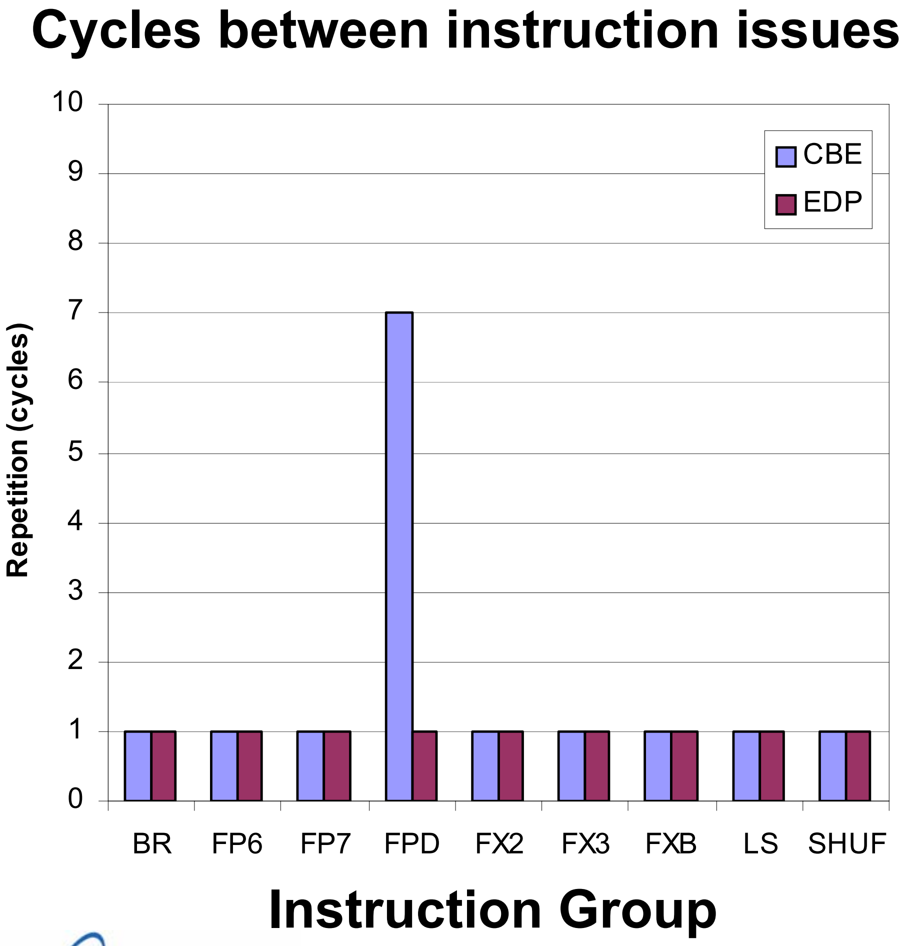
Computation: Cell-BE eDP has much improved DP floating-point performance

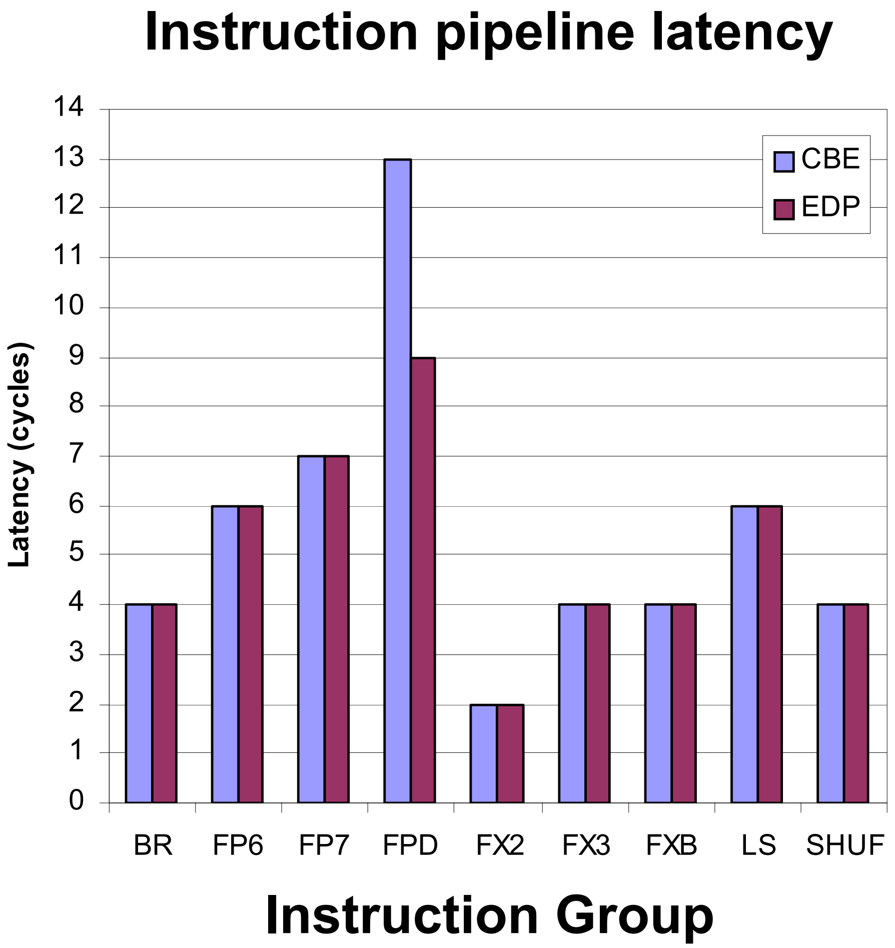
- Cell-BE had low DP floating-point performance
- Cell-BE eDP increased peak DP by 7x, and uses DDR2 memory
- PAL tested eDP (July ’07 and Sep ’07):
  - summary of testing from Sep with two memory speeds (667MHz and 800MHz)

	eDP-667 vs. CBE	eDP-800 vs. CBE
VPIC	1.01x	1.01x
CellMD	1.50x	1.50x
Hybrid-IMC	1.50x	1.50x
PAL-Sweep3D	1.72x	1.77x

- eDP available today in the IBM on-Demand center

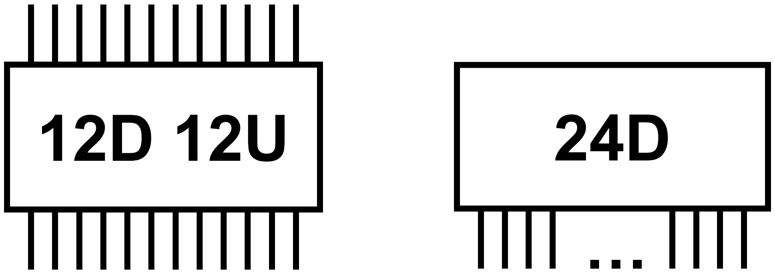
Cell-BE eDP vs. Cell-BE instruction costs



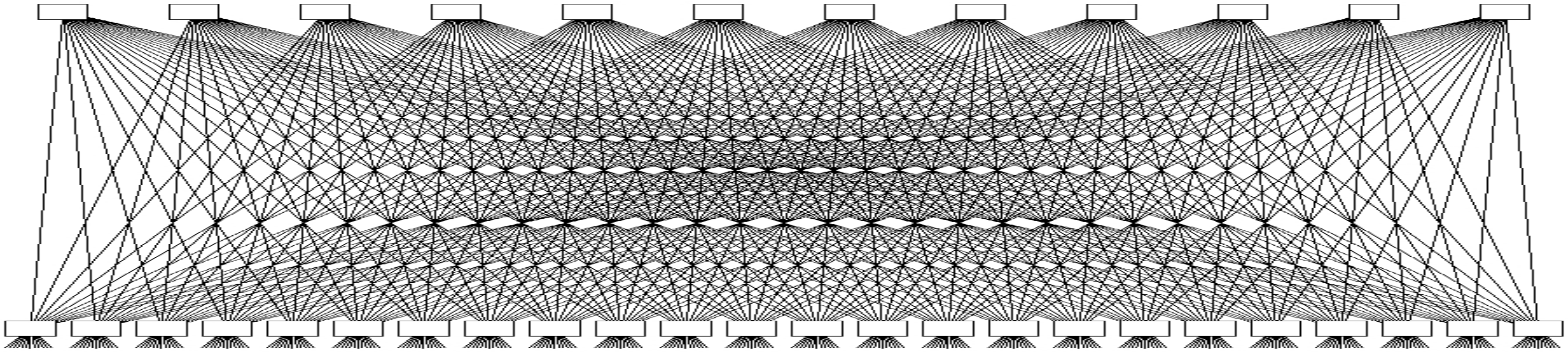


Infiniband Network Characteristics

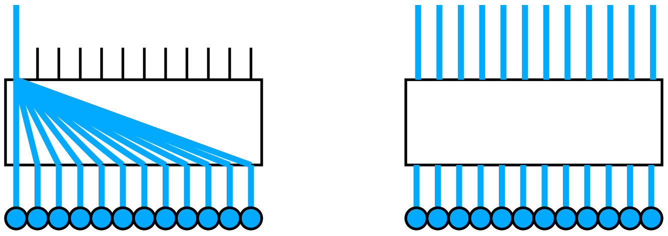
- Building block is a 24-port x-bar switch, e.g.



- Tree networks, e.g. 2-level, 288 port switch:

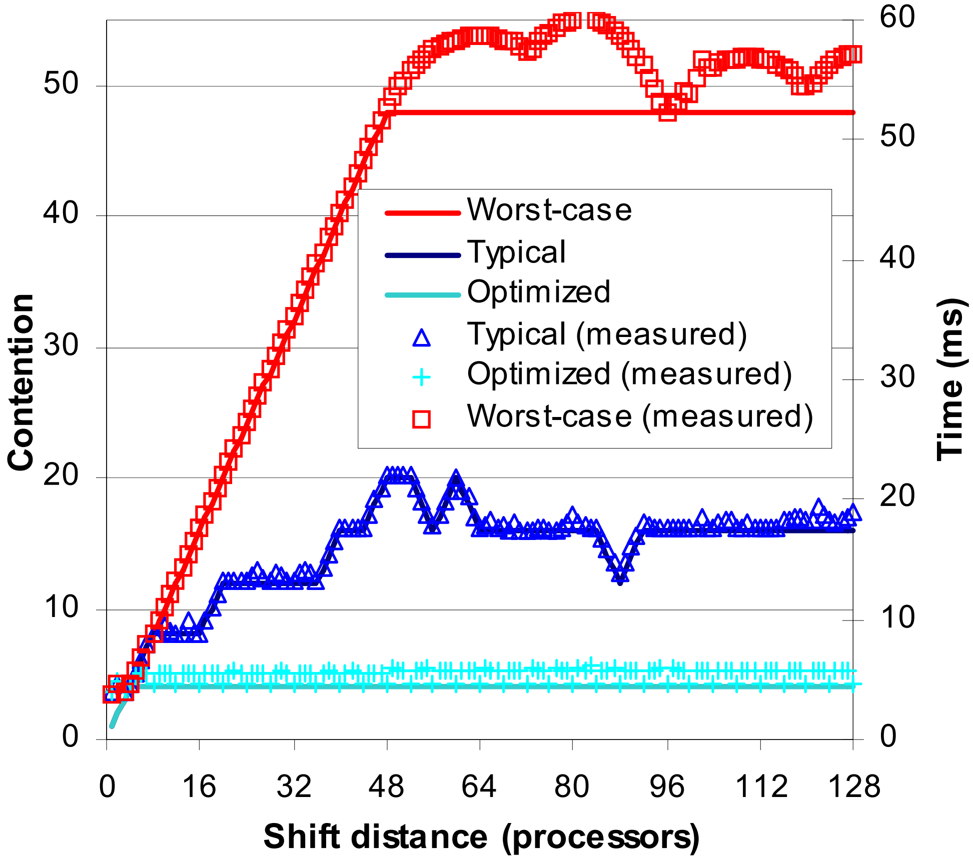


- Routing table in each switch determines output port for a message based on destination



Optimization produced increased network performance

- **Use logical-shift communication pattern**
  - P_i -> $P_i + d$ where d = 1..128
- **Maximum contention plotted (1024 PE job)**



- **Worst-case: max of 48 (# PEs attached to 1 switch)**
- **Typical: contention generally increases with shift distance**
- **Optimized: max of 4 (bottleneck is node-size, PEs)**

Application 1: VPIC

- **Plasma Particle-in-Cell,**
  - Cell-centric on Roadrunner, Opterons used for message relay
- **3-D volume containing ions and electrons**
  - Split into Voxels
  - Each voxel contains an ~equal number of ions+electrons
  - ions and electrons can move
    - **Results in inter-processor communication**
- **Parallel Decomposition: in 1-D, 2-D or 3-D**
  - Weak-scaling: constant work per processor
- **Two main model components**
  - Time to process a single ion/electron
    - **found to be same for both particle species**
  - size, pattern and number of communications per iteration
    - **1-D, 2-D or 3-D pattern depending on decomposition**

PAL

VPIC: Model Input Parameters

Input-deck	Hot
Parallel Decomposition	3D
Voxels / processor	16x16x16 (= 4K)
Particles / Voxel	512
# Particle species	2
Total # Particles / processor	4 M
Particle Size (for communications)	44B

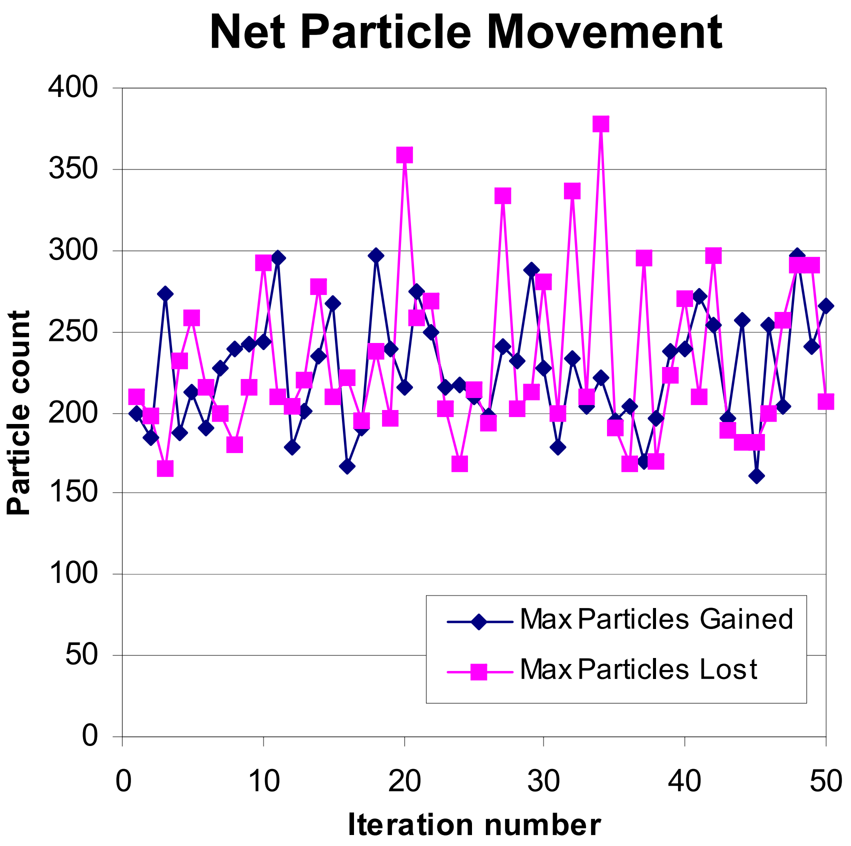
- **Compute performance per particle**

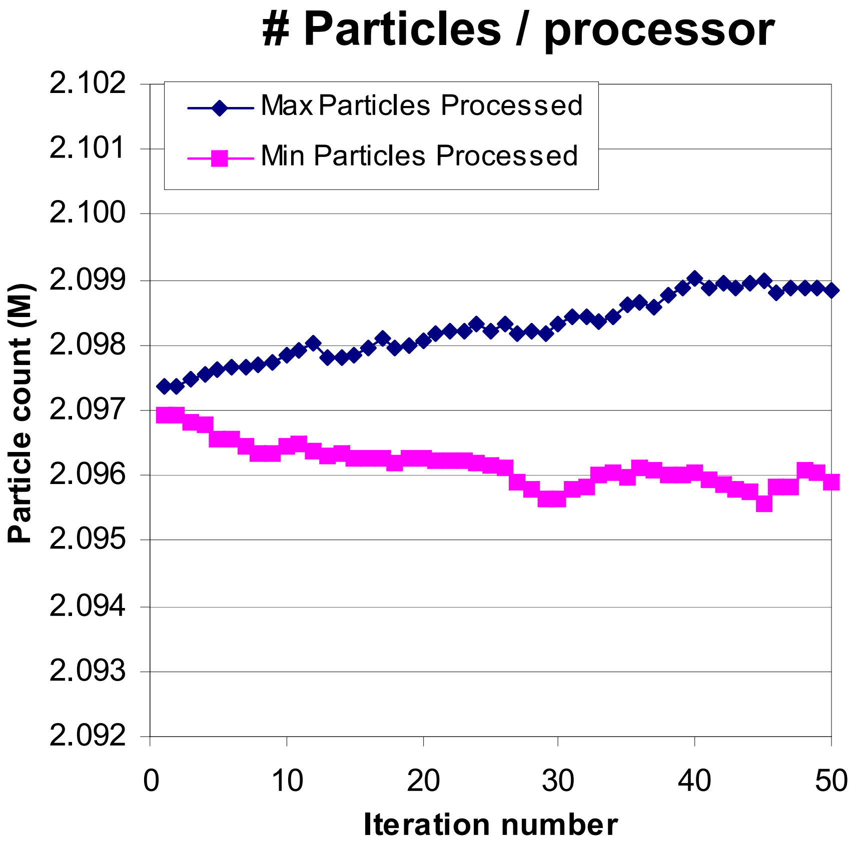
	Cell-eDP only	Opteron only
Compute per Particle	13ns	76ns

- **Note: Compute time is a composite of all stages**
  - On the Cell: main component (particle-push) done on all SPEs
  - Some steps including sorting currently on the PPE
    - **Electron sorting every 50 iterations, and ion sorting every 100 iterations**

VPIC: Compute Considerations

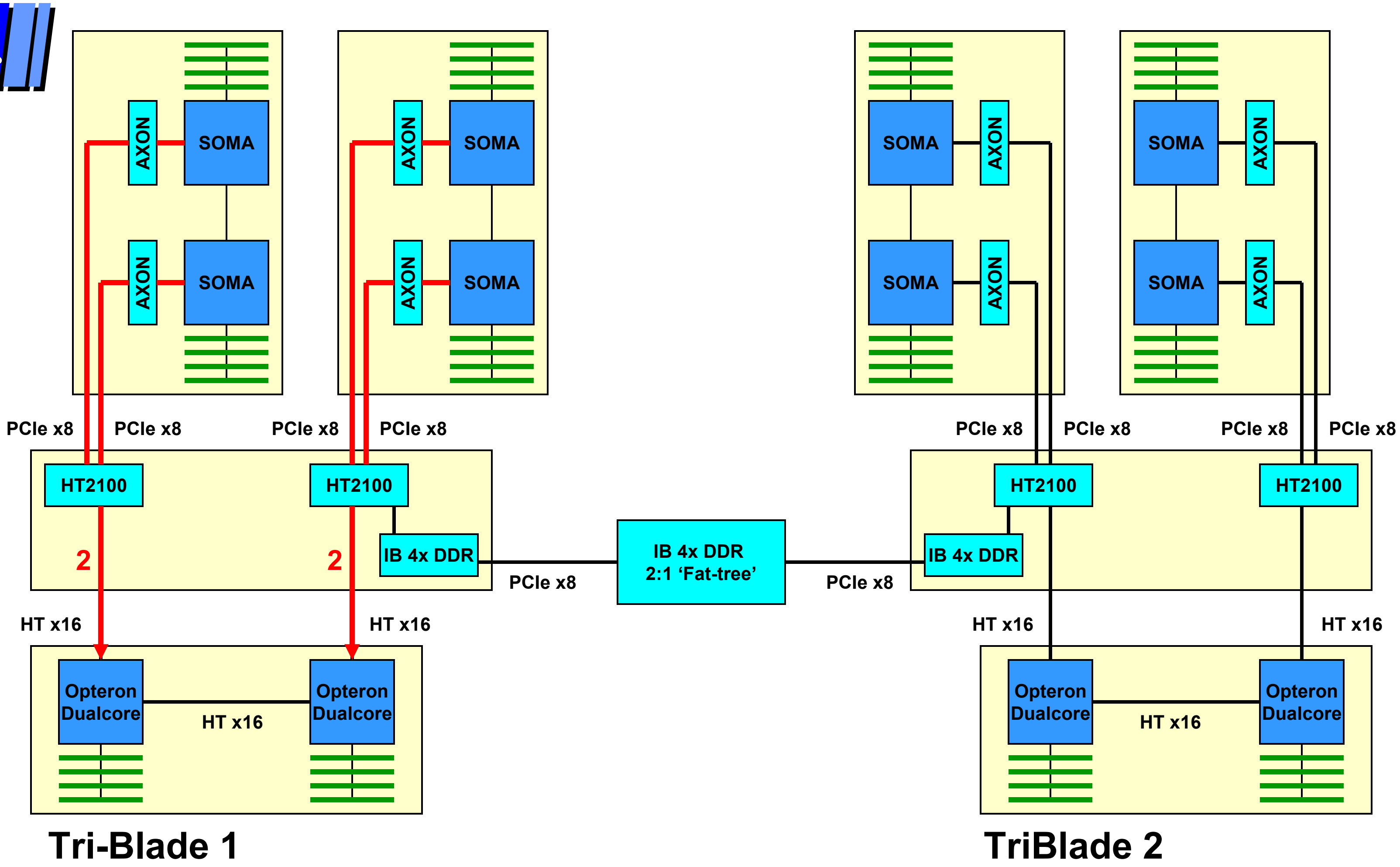
- **# particles per processor can vary over iterations**
  - Input deck dependent





VPIC: Parallel Aspects

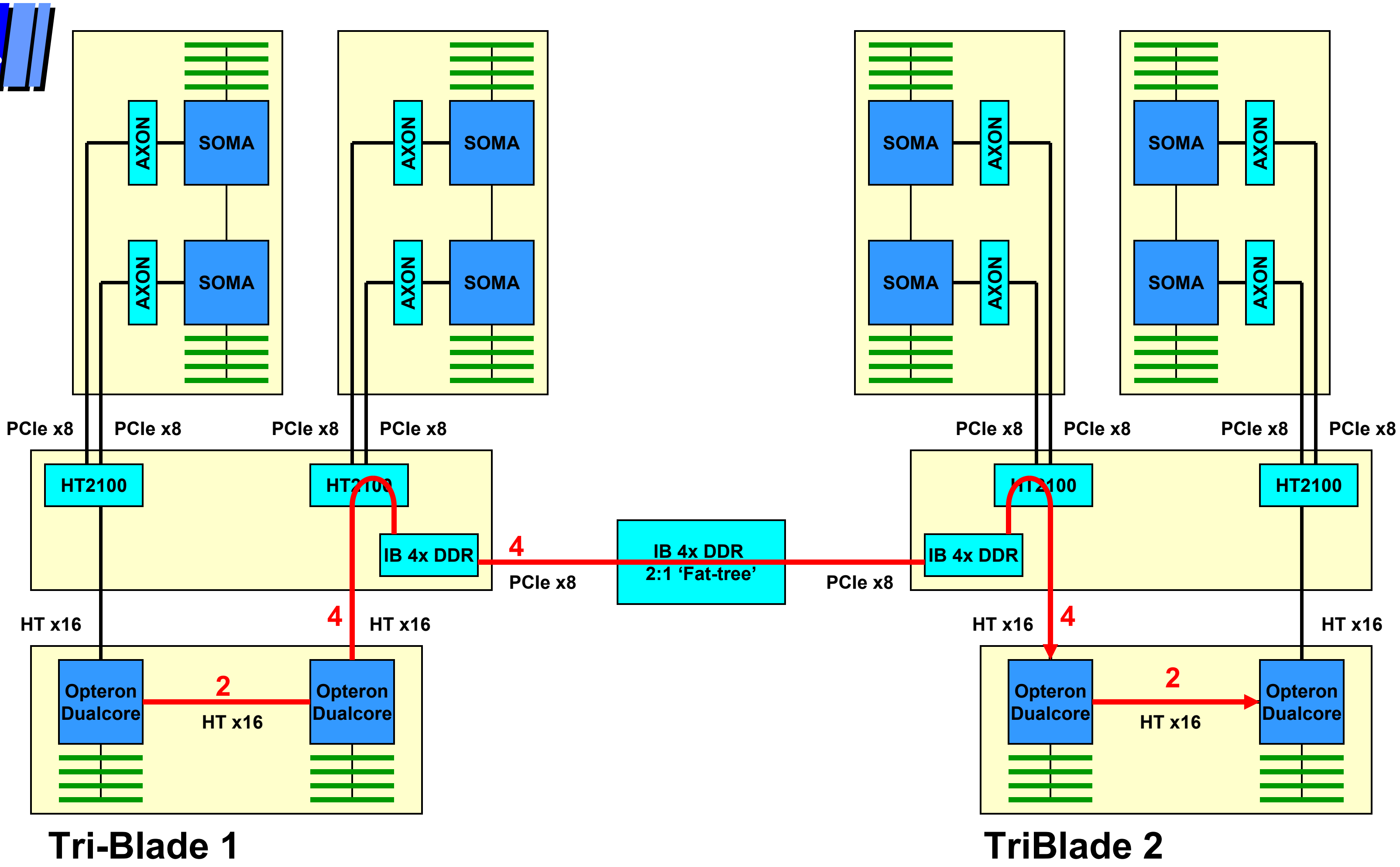
- **Assumed linear MPI rank mapping to nodes**
  - Rank 0-3 on first triblade, Rank 4-7 on second etc.
- **Communications take place in each of 6 directions:**
  - Particle transfer:
    - **One message per neighbor per iteration per species**
    - **4-10KB (ion movement), ~20-45KB (electron movement)**
  - Remaining messages are small: 4B
  - Total of 23 messages per neighbor per iteration
- **Model initially developed for non-accelerated VPIC**
  - Validated with high accuracy on 1024core AMD IB cluster
- **Refined for hybrid implementation with message relay**
  - Model accuracy within 5% on available AAIS hardware (8 blades)



→ 1) Cells (TriB 1) -> Opterons (TriB 1)

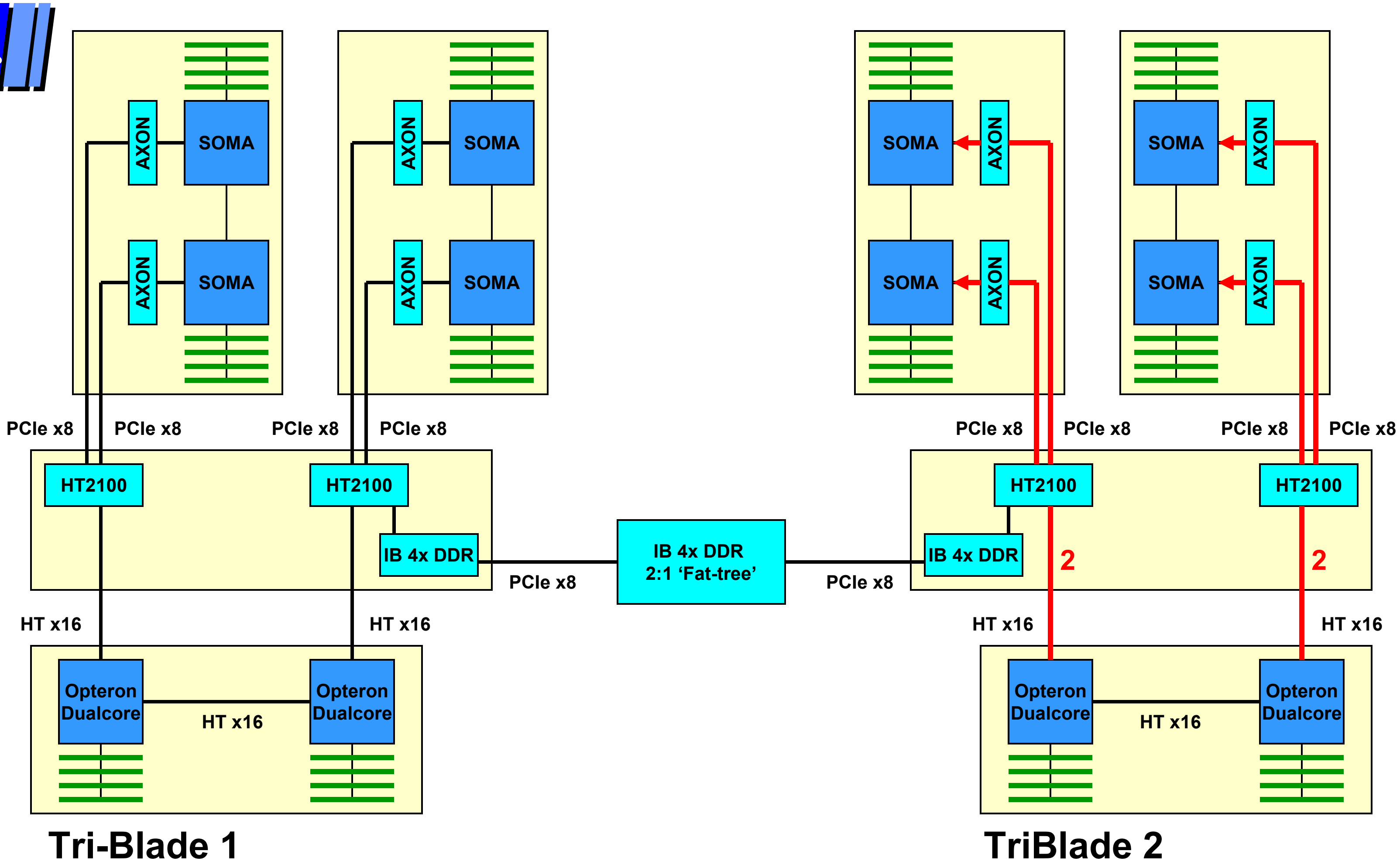
2) Opterons (TriB 1) -> Opterons (TriB 2)

3) Opterons (TriB 2) -> Cells (TriB 2)



1) Cells (TriB 1) -> Opterons (TriB 1)
2) Opterons (TriB 1) -> Opterons (TriB 2)
3) Opterons (TriB 2) -> Cells (TriB 2)

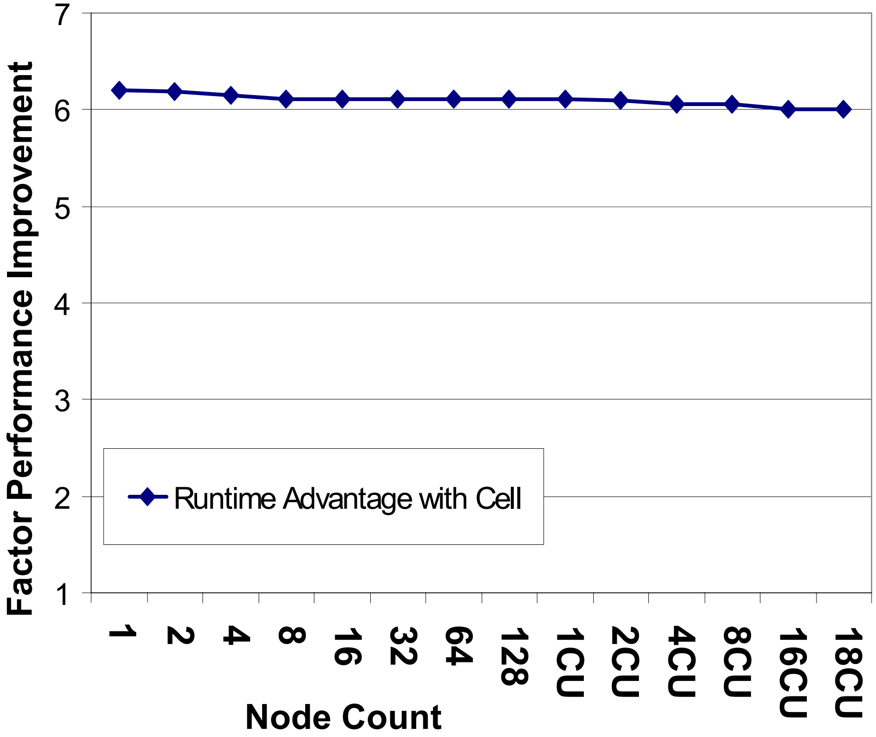
PAL



1) Cells (TriB 1) -> Opterons (TriB 1)
2) Opterons (TriB 1) -> Opterons (TriB 2)
3) Opterons (TriB 2) -> Cells (TriB 2)

VPIC: RR Performance predictions

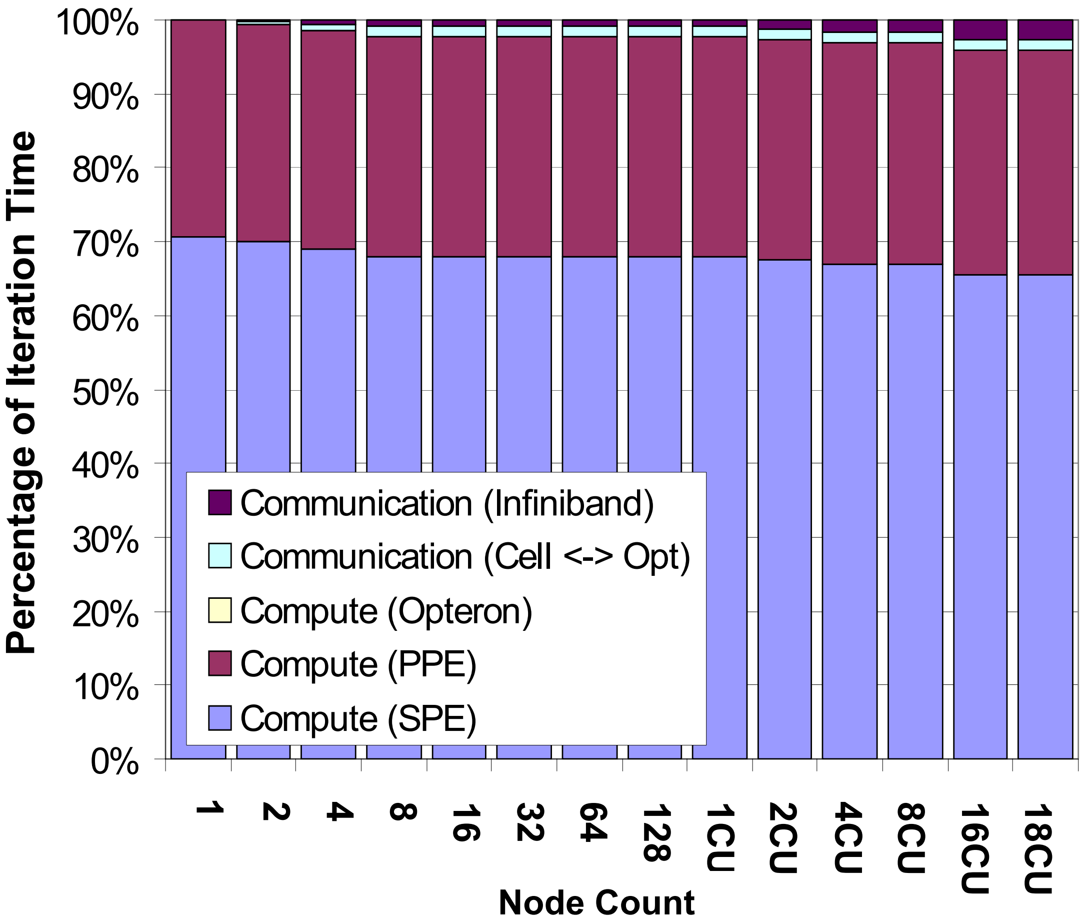
Runtime on Opterons / Runtime on accelerated RR



- Very Good scaling expected
- With current code, expect a factor of ~6x better performance using Cell

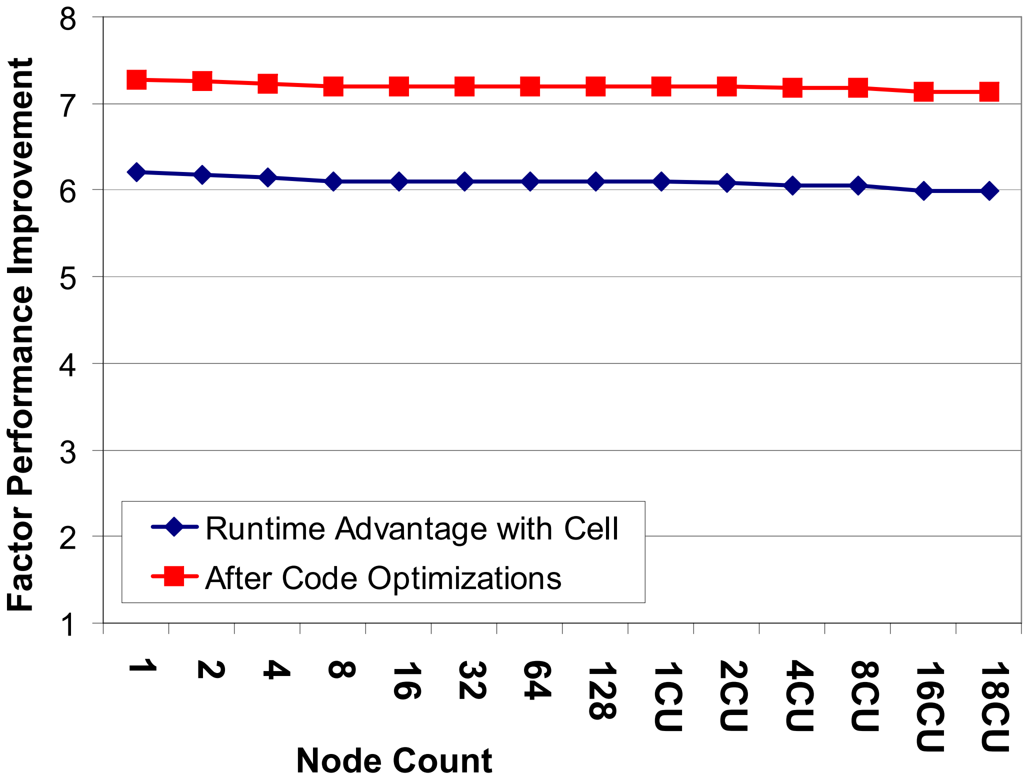
VPIC: Profiling

- **Where is the time being spent ?**
  - Remains compute bound
  - ~65% SPU
  - ~31% PPU
  - ~1 Cell <-> Opteron
  - ~3% Infiniband



VPIC: Possible Code Improvements

- **Between now and RR deployment expect:**
  - Migration of particle sort from SPU to PPU (x0.5)

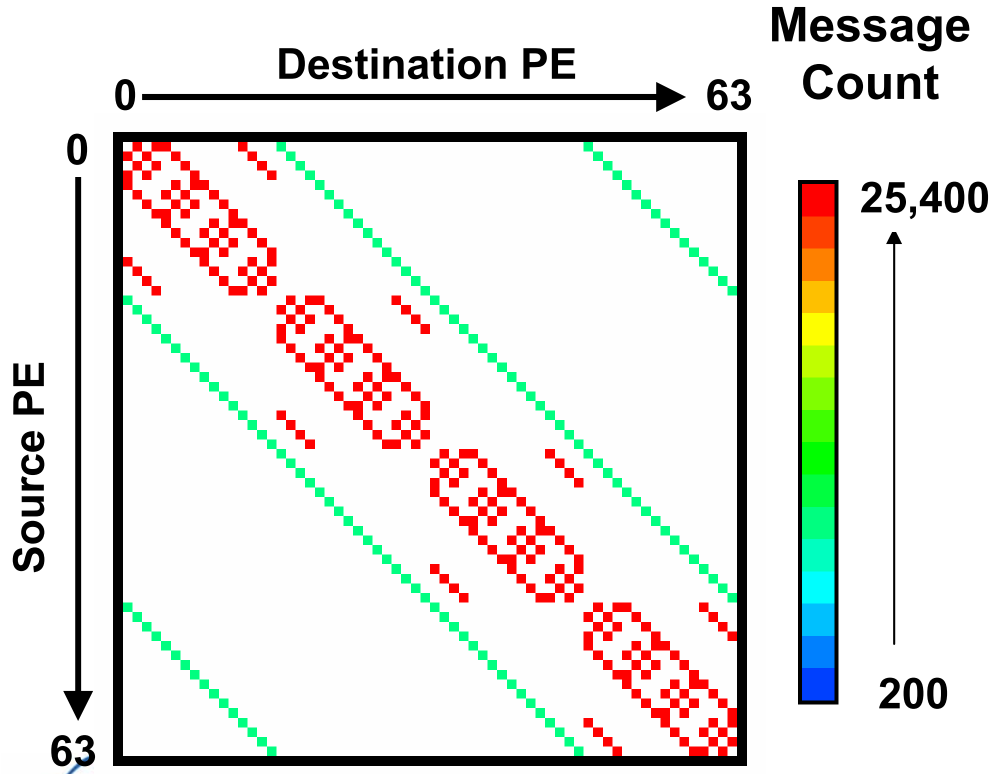


Application 2: SPaSM

- **Single species of atoms arranged in 3-D structure**
  - uniform spatial distribution (crystalline structure, possible voids)
  - uniform, very short range interactions
- **Three types of cell !**
  - Unit cell – defining the atom structure
  - Computation cell – defining a ‘unit’ of work
  - Processor cell – doing a lot of the work !
- **3-D structure partitioned in 3-D, 26 neighbors**
  - computational cells are carefully ordered to minimize communications
- **Approach:**
  - Understand and model existing SPaSM code
  - Validate model on existing cluster hardware
  - Predict performance on Roadrunner
- **Existing code**
  - very different performance characteristics to Roadrunner code
  - lots of small messages, one per boundary computation-cell

SPaSM: Communication Pattern

- **Example: Communication summary (one iteration)**
  - 4x4x4 processors, 512x512x512 unit-cells
  - Does not show temporal information



- **Diagonals indicate:**
  - ±X, ±Y, ±Z comm. directions
  - Also cycle boundaries
- **Each diagonal is a logical "shift" of a certain distance**
- **Detailed analysis reveals:**
  - #messages/PE = 120K
  - Half are of size 56B
  - Other half range in size from 4x536B to 14x536B

SPaSM Workload Characteristics (Sep'07)

- **Hybrid accelerator Approach**
- **Acceleration of major part of processing**
  - Accelerated 90% of original microprocessor cycles
- **Processing flow (an iteration):**
  - Prepare data on AMD for Cell
  - Transfer data volume to Cell (~230MB)
  - Process data on Cell
  - Transfer data volume back to AMD (~230MB)
  - Post-process on AMD
  - Update Particles on AMD
  - Exchange boundaries between AMDs (~250MB total in 6 messages)

SPaSM: Model Input Parameters

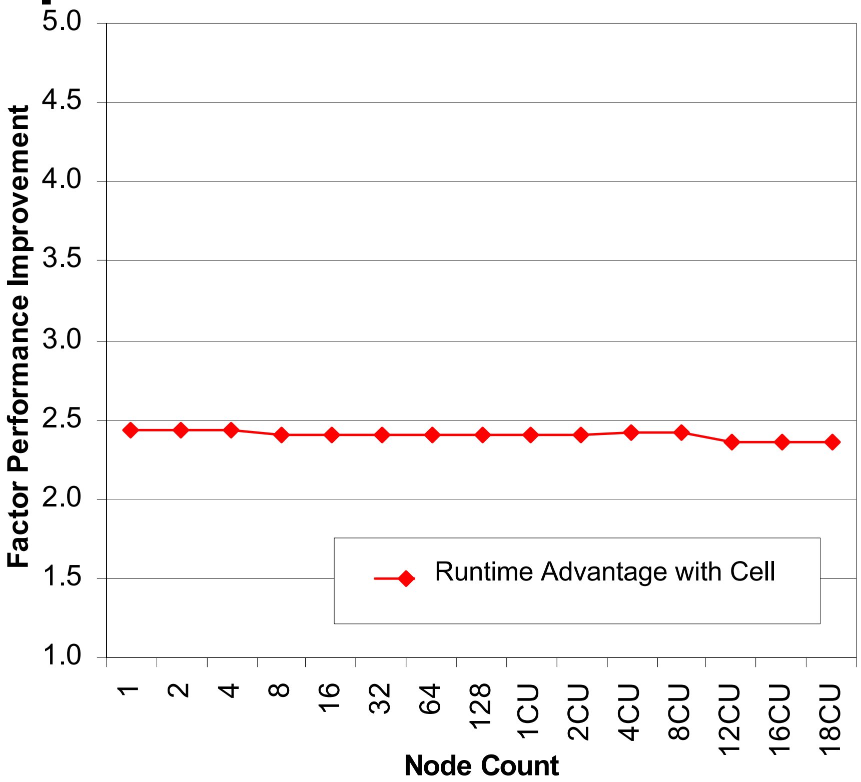
- **Weak Scaling: Problem size fixed at 1.5M atoms per processor**
  - 64x64x64 unit-cells x 6 particles/unit-cell)
- **Iterative**

  - Compute-time per iteration varies very little (max. of a few percent)

Input-deck	R2
Unit cells / processor	64x64x64
Computational cells / processor	46x46x69
Av. Atoms / c-cell	10.8
Skin Depth	2
Size of particle (Node <-> Node)	590B
Size of particle (Cell <-> Opteron)	132B
Compute per atom (Opteron component)	1.23μs
Compute per atom (Cell-eDP component)	2.7μs

SPaSM: RR Performance predictions

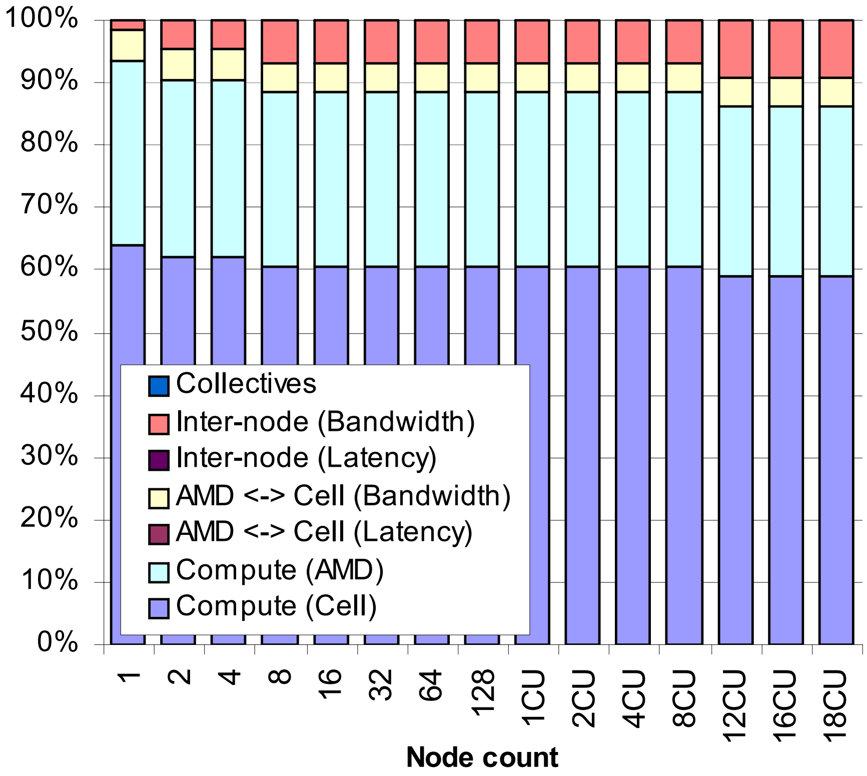
Runtime on Opterons / Runtime on accelerated RR



- Very Good scaling expected
- With Sep’07 code, expected a factor of ~2.4x better performance using the Cell

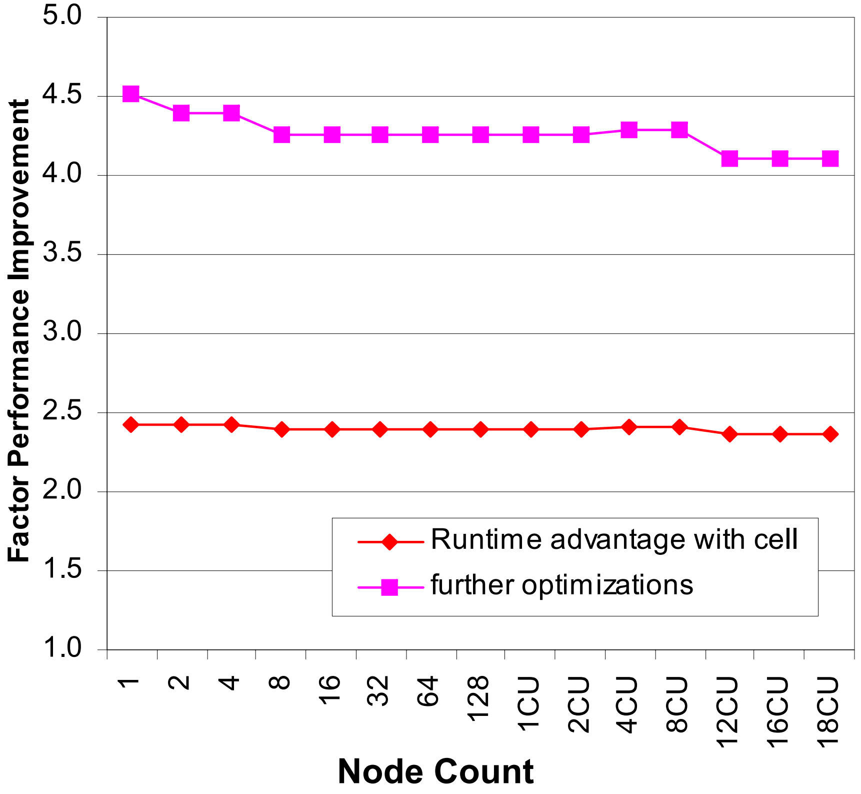
SPaSM: Profiling

- **Where is the time being spent ?**
  - Remains compute bound
  - ~60% time on the Cell
  - ~26% time on Opteron
  - ~9% in Infiniband
  - ~5% in Cell <-> Opteron



SPaSM: Possible code improvements

- **Between now and RR deployment expect:**
  - Improvement of cell computation (reduction of neighbors) (x0.6)
  - Improvement on AMD side (x0.3)



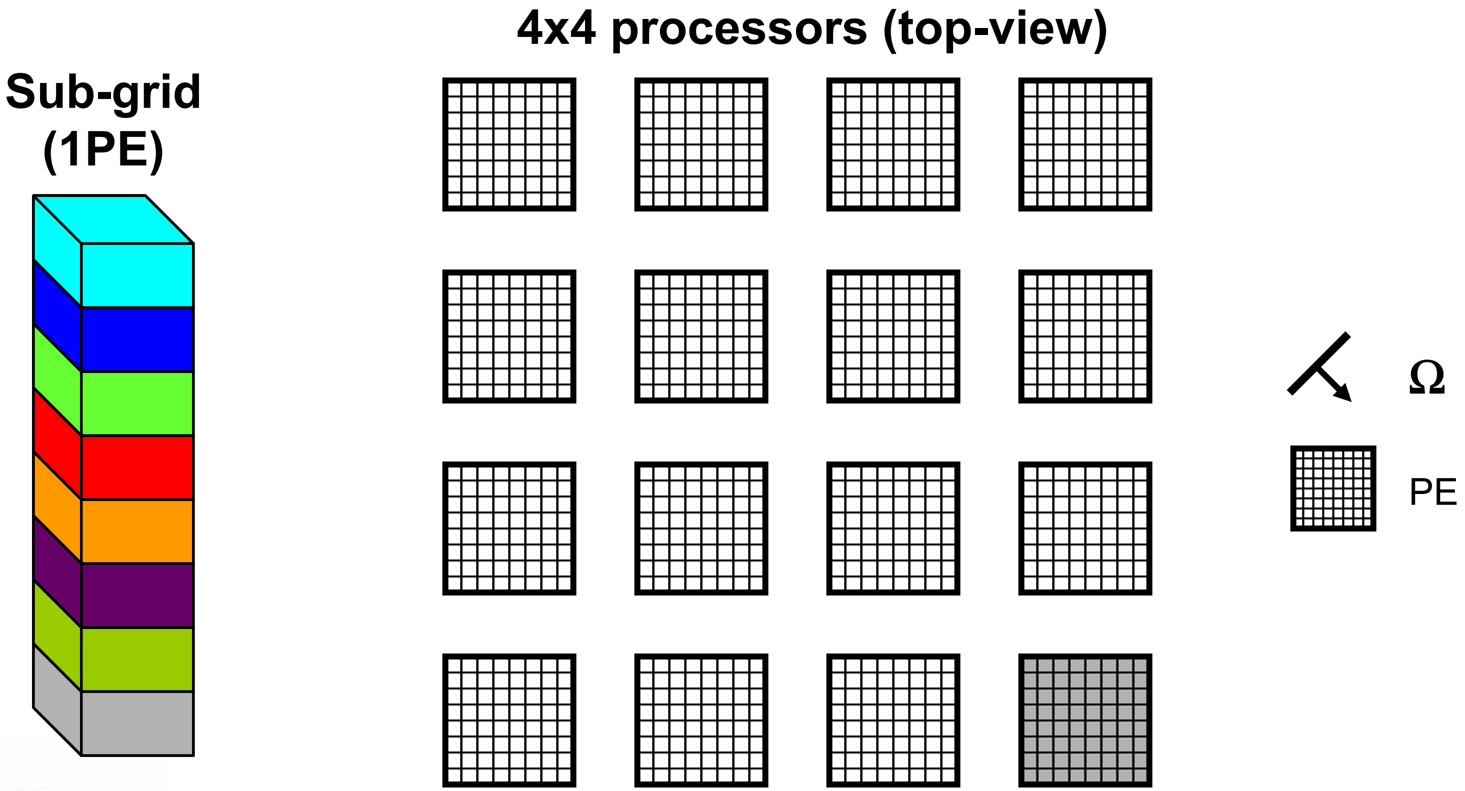
Application 3: Sweep3D Input Parameters

- **PAL optimized version of Sweep3D for Cell**
- **Uses domain decomposition (in 2-D)**
  - Each SPE processes a defined subgrid
  - 32 subgrids per triblade
- **A key parameter is the computational block size**
  - Angles per block fixed at 6 (for high SPE compute efficiency)
  - K-planes per block is variable (decreases with scale for high parallel efficiency)

Sub-grid size per SPE (I x J x K)	**5x5x400**
K-planes per block	**{1 .. 50}**
Angles per block	**6**
Number of cycles	**10**
Grind time per grid-point per angle (eDP) **NB variable depending on block-size**	**{29 .. 47} ns**
Boundary surface (Bytes per grid-point per angle)	**8**

Wavefront Parallelization

- Pipeline characteristic whose length increases with scale
- 3-D grid is typically parallelized in only 2-D
  - Blocking used to increase parallel efficiency (c.f. blocking for cache)

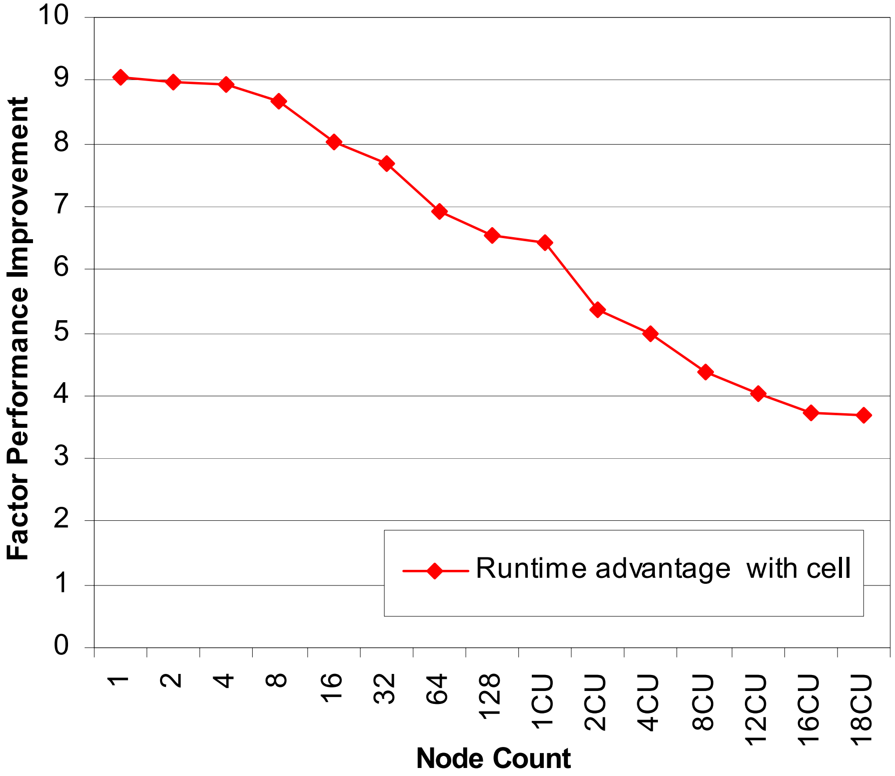


Sweep3D Workload Characteristics

- **Mapping of Sweep3D to the Triblade**
  - Processing
    - **Cell – SPU: main sweep processing**
    - **Cell – PPU: DMA and inter-SPE communication management**
    - **Opteron: No computation**
  - Message Passing: Originate on the Cell and relayed through Opterons
- **Message characteristics**
  - Fine-grained communications:
    - **2 messages sent per SPE per block per cycle**
    - **Sizes depend on block size, 240B -> 4,800B (typical)**
- **At small-scale performance is compute-bound**
- **At large-scale performance is impacted by both message latency and increased pipeline length**
- **Performance Model validated on all large-scale systems**
- **Model adapted to reflect additional Cell->AMD communications**

Sweep3D: RR Performance predictions

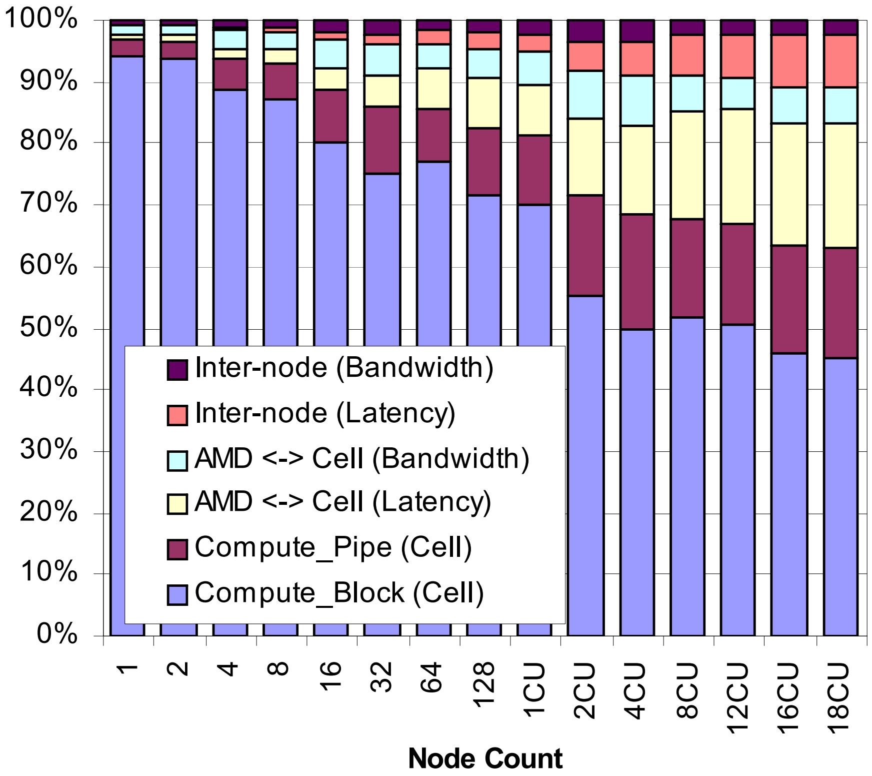
Runtime on the base cluster / Runtime on accelerated RR



- **Sweep3D sensitive to latency**
  - Increased due to Cell <-> Opteron
  - But some communication can be overlapped
- **Performance advantage of accelerator reduces with scale**

Sweep3D: Profiling

- **Where is the time being spent ?**
  - ~63% Compute on Cell
  - ~20% Latency (Cell <-> AMD)
  - ~5% Bandwidth (Cell <-> AMD)
  - ~8% Latency (Infiniband)
  - ~3% Bandwidth (Infiniband)



- **Pipeline unavoidable**
- **Latency dominates communication (Cell <-> AMD is major component)**

Comparison to ASCI Q

- ASCI Q was the largest machine in use at LANL until recently
- 4-processor (Alpha) EV68 nodes interconnected by Quadrics QSNet-1.
- Peak speed of 20 Tflops
- Comparison made to insert a “historical” perspective in the analysis

Runtime improvement of RR vs. ASC Q (equal node-count basis)

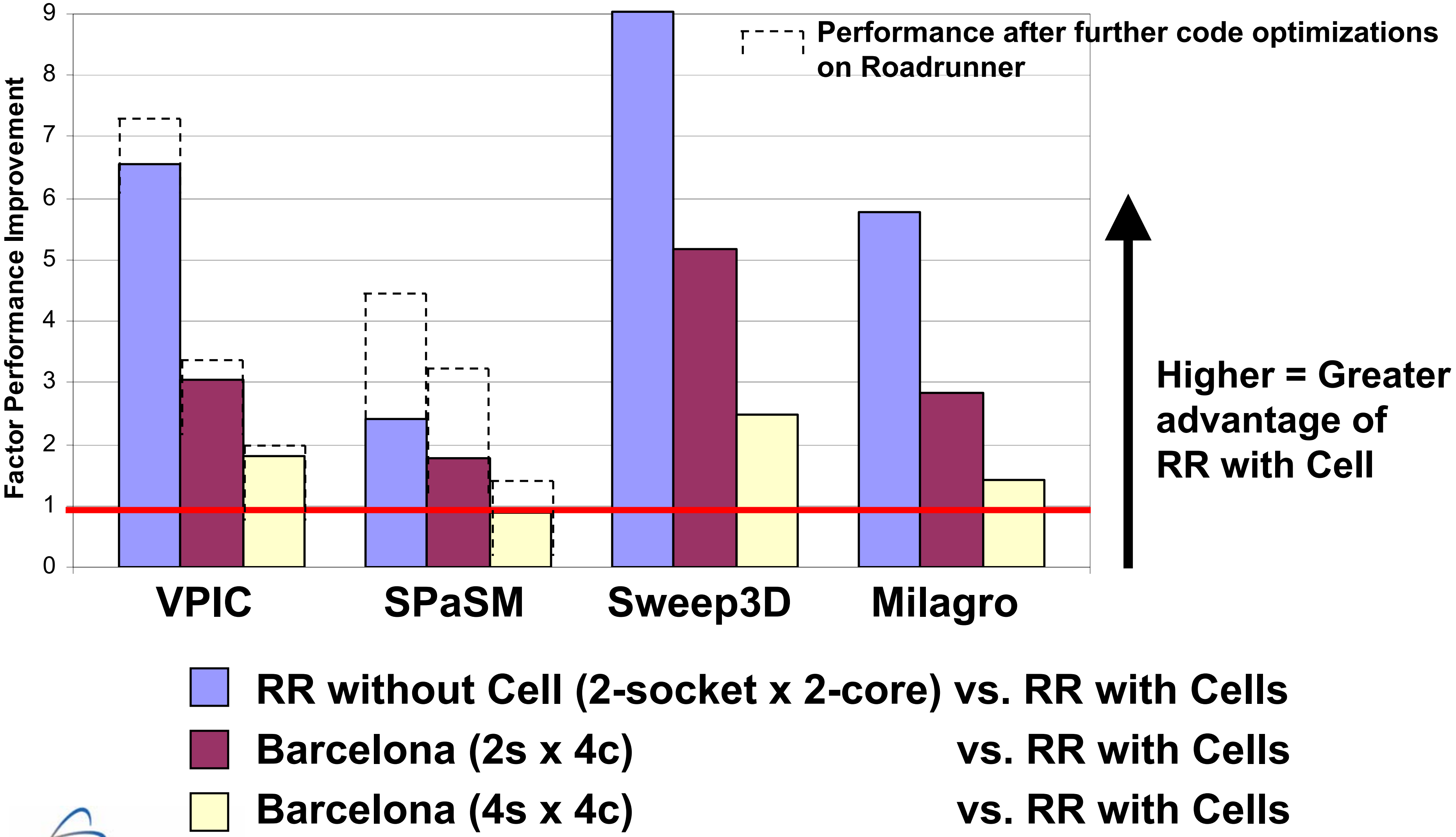
	1 Node	At Scale
VPIC	23	31 (800 Nodes)
SPASM	4.5	5 (256 Nodes)
Sweep3D	16	15 (810 Nodes)
Milagro	9	12 (800 Nodes)

Roadrunner Performance Relative to other (Hypothetical) Systems

- **Nodes used for comparison:**
  - Triblade (4x cell-eDP, and AMD 2-socket x 2-core) [Roadrunner]
  - AMD Barcelona 2-socket x 4-core (2GHz)
  - AMD Barcelona 4-socket x 4-core (2GHz)

- **Fixed problem size per node**
  - when comparing node performance

Single Node Performance Comparison



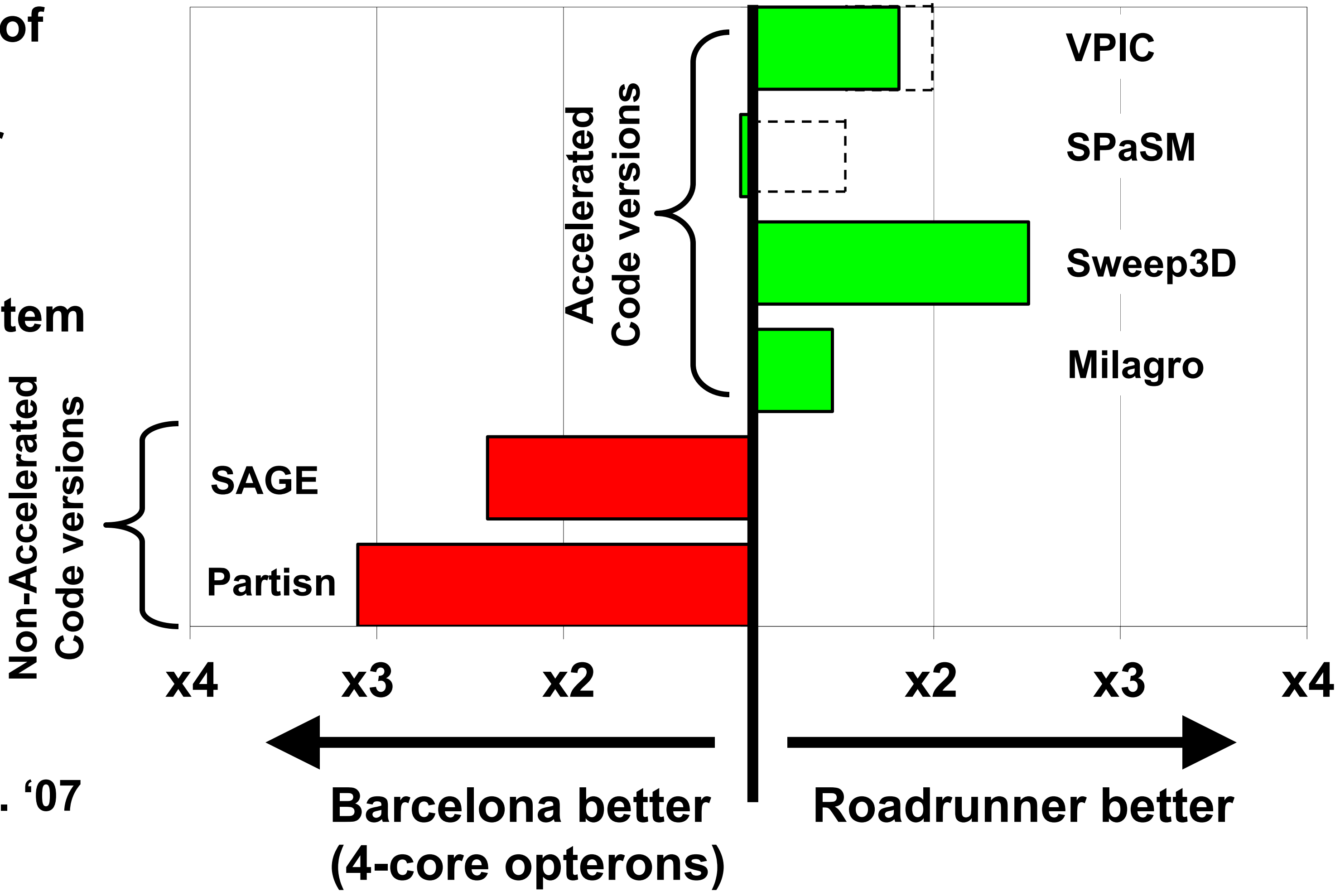
Performance after further code optimizations on Roadrunner

Higher = Greater advantage of RR with Cell

Results: Roadrunner has a significant performance advantage

Performance of

Roadrunner
vs.
equivalent
Quad-core System



Note:
Codes as of Sept. ‘07

Summary

- Analyzed RR performance under a realistic application workload of interest through predictive modeling
- VPIC, SPaSM and Sweep3D scale well on RR
- VPIC, SPaSM, Sweep3D exhibit high performance gains over the RR base cluster
  - in the range of 2.5x-7x
- Significant performance improvements over ASC Q
- Accelerated applications under consideration are faster on RR than on hypothetical systems using state-of-the-art multicore nodes

Achievements

- **Performance analysis and predictions at scale**
- **Optimized Network routing for improved performance**
- **Cell Messaging Layer (CML)**
  - Developed from PAL's implementation of Sweep3D
  - Each SPE has a separate MPI rank in CML and can communication with any other SPE in the system
  - Open sourced, peer reviewed paper at IPDPS, April 2008
- **JumboMem**
  - Enables a single process to use memory throughout a cluster
  - Transparent to an application
  - For RR – the Cells can use the Opteron memory (or vice-versa) [under-development for the triblades]